Couplings between neutrinos and fuzzy dark matter in GLoBES

Vedran Brdar, Joachim Kopp, Jia Liu, Xiaoping Wang

v1.0 (June 2019)

1 Introduction

If the dark matter (DM) in the Universe couples to neutrinos, the propagation of neutrinos through the omnipresent DM background can affect oscillations. This is similar in spirit to Mikheyev-Smirnov-Wolfenstein-type matter effects in the Standard Model: neutrinos can undergo coherent forward scattering on the background particles, i.e. they can interact with the background without changing their quantum state, or the background's quantum state. It is therefore impossible to tell which background particle a given neutrino has interacted with, so all of them contribute coherently, leading to a huge enhancement factor. The enhancement factor is proportional to the number density of backgroud particles, and it is for this reason that neutrino–DM interactions are most easily observable for extremely light DM particles ($\lesssim 10^{-20}\,\mathrm{eV}$), which are called "fuzzy DM" because of their huge Compton wave length. Given the known mass density of DM, fuzzy DM must have a huge number density. Neutrino interactions with fuzzy DM have first been investigated in [1, 3, 2].

The present code makes this scenario accessible to GLoBES. It supports both, couplings of neutrinos to fuzzy DM in the form of new scalar particles, or in the form of new vector bosons. In the latter case, it is possible to specify the polarization of the DM relative to the neutrio beam direction, or to assume unpolarized DM. (It is not fully understood whether or not fuzzy vector DM would be polarized or unpolarized in the Milky Way.)

1.1 Scalar DM

For scalar DM, the relevant terms in the phenomenological Lagrangian are [2]

$$\mathcal{L}_{\text{scalar}} = \bar{\nu}_L^\alpha i\partial\!\!\!/ \nu_L^\alpha - \tfrac{1}{2} m_\nu^{\alpha\beta} \overline{(\nu_L^c)^\alpha} \nu_L^\beta - \tfrac{1}{2} y^{\alpha\beta} \phi \, \overline{(\nu_L^c)^\alpha} \nu_L^\beta \,. \tag{1}$$

The first term is the standard kinetic term, the second one is a standard Majoarana mass term, and the third one describes the couplings between the scalar DM particle ϕ and the left-handed SM neutrinos ν_L^α. The strength of the coupling is denoted by the complex symmetric matrix $y^{\alpha\beta}$. In all terms, the flavor indices α and β run over e, μ, or τ, and ν_L^c refers to the charge conjugated field.

The DM field ϕ in eq. (1) can be written as

$$\phi = \bar{\phi}\cos(m_\phi t)\,, \tag{2}$$

with the DM mass m_ϕ and the normalization

$$\bar{\phi} = \frac{\sqrt{2\rho_\phi}}{m_\phi}\,. \tag{3}$$

In the last expression, $\rho_\phi \simeq 0.3\,\mathrm{GeV/cm^3}$ is the DM energy density at the location of the Earth (~ 8 kpc from the Galactic Center). Equation (2) describes a slowly varying function of time, with a periodicity $T = 2\pi/m_\phi = 4.77\,\text{days} \times (10^{-20}\,\mathrm{eV}/m_\phi)$.

In view of eq. (3), we define the *effective DM couplings*

$$\chi^{\alpha\beta} \equiv y^{\alpha\beta}\frac{\sqrt{2\rho_\phi}}{m_\phi}\,. \tag{4}$$

We see from eq. (1) that neutrino couplings to scalar DM can be viewed as a dynamic modification to the neutrino mass matrix. This is exactly how they are implemented in `fuzzy-dm.c`: the code reconstructs the vacuum mass matrix from the given parameters, adds the DM contribution, and then re-diagonalizes the result to obtain an effective PMNS matrix and effective mass squared differences.

1.2 Vector DM

The Lagrangian for DM couplings to fuzzy vector DM is [2]

$$\mathcal{L}_\text{vector} = \bar{\nu}_L^\alpha i\partial\!\!\!/\nu_L^\alpha - \tfrac{1}{2}m_\nu^{\alpha\beta}\overline{(\nu_L^c)^\alpha}\nu_L^\beta + gQ^{\alpha\beta}\phi^\mu\bar{\nu}_L^\alpha\gamma_\mu\nu_L^\beta\,, \tag{5}$$

The first two terms, which appear also in the standard scenario, are the same as in eq. (1) above; the last term has the form of a gauge coupling with gauge boson ϕ^μ and flavor-dependent (complex hermitian) charges $Q^{\alpha\beta}$. We write ϕ^μ as

$$\phi^\mu = \bar{\phi}\xi^\mu\cos(m_\phi t)\,, \tag{6}$$

where ϕ^0 is the same as in eq. (3). The polarization vector ξ^0 describes the alignment of the DM field relative to the beam, where the beam is always assumed to travel in the positive z-direction. The effect of the DM–neutrino coupling for vector DM is to modify the neutrino dispersion relation into

$$E^2 = p^2 + m_\nu^2 \qquad \to \qquad (E + gQ\xi^0\bar{\phi})^2 = (\vec{p} + gQ\vec{\xi}\bar{\phi})^2 + m_\nu^2\,. \tag{7}$$

(Here, we have for simplicity dropped the time-dependent cosine functions.) The code uses eq. (7) to compute the modified Hamiltonian for neutrinos propagating through the

DM background. Specifically, the Hamiltonian turns into

$$\hat{H} = \frac{1}{2E} U \begin{pmatrix} 0 & & \\ & \Delta m_{21}^2 & \\ & & \Delta m_{31}^2 \end{pmatrix} U^\dagger + \frac{1}{2E} g^2 \bar{\phi}^2 \vec{\xi}^2 Q^\dagger Q - g Q \bar{\phi} p^\mu \xi_\mu \,. \tag{8}$$

The last term is relevant only for polarized vector DM – for unpolarized DM, it averages to zero when ξ^μ is averaged over. All terms are replaced by their complex conjugates when considering oscillations of antineutrinos. The last term also changes sign for antineutrinos.

2 Initializing the Oscillation Engine

To use the code, include `fuzzy-dm.c` in your project by modifying your `Makefile` accordingly, and `#include` the header file `fuzzy-dm.h` in your source code.

The first step is to initialize the oscillation engine using the function

```
int dm_init_probability_engine(int dm_type);
```

The argument `dm_type` can be `DM_SCALAR` for scalar DM, `DM_VECTOR_POLARIZED` for polarized vector DM, or `DM_VECTOR_UNPOLARIZED` for unpolarized vector DM. These three constants are defined in `fuzzy-dm.h`. The DM type can be changed later using

```
int dm_set_dm_type(int dm_type);
```

and it can be queried using

```
int dm_get_dm_type();
```

The oscillation engine knows of course the six standard oscillation parameters (named `TH12`, `TH13`, `TH23`, `DELTA_0`, `DM21`, and `DM31`), but it also defines the following additional parameters (in the given order):
It is strongly recommended to use `glbSetParamName` to assign human-readable names to the oscillation parameters by which they can be referred to in subsequent calls to `glbSetOscParamByName` and `glbSetProjectionFlagByName`. The pre-defined names listed above can be retrieved using

```
const char *dm_get_param_name(const int i);
```

3 Making the Oscillation Engine Known to GLoBES

After initializing the fuzzy DM oscillation, we need to tell GLoBES that we want to used it by calling

```
glbRegisterProbabilityEngine(n_params,
                             &dm_probability_matrix,
                             &dm_set_oscillation_parameters,
                             &dm_get_oscillation_parameters,
                             NULL);
```

The number of oscillation parameters `n_params` in the first line can in principle be deduced from the above table, but we highly advice not to hardcode it, but to instead retrieve it by calling

```
int dm_get_n_osc_params();
```

before calling `glbRegistareProbabilityEngine` (but after calling `dm_init_probability_engine`.

As with any new probability engine, the call to `glbRegisterProbabilityEngine` has to occur before any calls to `glbAllocParams` or `glbAllocProjection` to make sure that all parameter and projection vectors have the correct length.

Having initialized and registered the non-standard probability engine, we can use all GLoBES functions in the same way as we would for standard oscillations.

After using the fuzz DM oscillation engine, we may want to release the (small) amount of memory allocated by it by calling

```
dm_free_probability_engine();
```

4 Time Dependence

Simulating the time dependence appearing in eqs. (2) and (6), as well as a possible time dependence in the polarization vector ξ^μ coming from the motion of the Earth through the DM fluid, is not intrinsically part of this package. The reason is that it can be easily implemented by varying $\bar{\phi}$.

A possible strategy for including the time dependence is to run the GLoBES simulation and fit multiple times for different time bins (with suitably scaled `@time` parameter in the flux definition). As the data in the different time bins are independent, the total χ^2 from such an analysis can be obtained simply by adding the χ^2's from all time bins.

If running the full simulation for sufficiently many time steps is too computationally expensive, the following approximation should work: use only few time bins for the main simulation, but when analyzing it (for instance in a user-defined GLoBES χ^2 function), divide it into finer time bins, interpolating between the coarse ones with a suitable chosen oscillatory fit function (for instance a polynomial in $\cos(m_\phi t)$). The parameters of the fit function typically do not need to be determined separately for each energy bin because they very slowly with energy. It is instead sufficient to carry out the fit in only a few energy bins and then interpolate the fit parameters in between. If more accuracy in the time domain is desired, one could imagine evaluating oscillation probabilities with finer time binning for just those few bins.

References

[1] Asher Berlin. Neutrino Oscillations as a Probe of Light Scalar Dark Matter. *Phys. Rev. Lett.*, 117(23):231801, 2016.

[2] Vedran Brdar, Joachim Kopp, Jia Liu, Pascal Prass, and Xiao-Ping Wang. Fuzzy Dark Matter and Non-Standard Neutrino Interactions. 2017.

[3] Gordan Krnjaic, Pedro A. N. Machado, and Lina Necib. Distorted neutrino oscillations from time varying cosmic fields. *Phys. Rev.*, D97(7):075017, 2018.

M1	the lightest neutrino mass eigenstate (which becomes relevant for neutrino couplings to scalar DM)
M_DM	the DM mass (currently unused)
ABS_CHI_$\alpha\beta$ ARG_CHI_$\alpha\beta$	the absolute values and phases of the effective DM–neutrino couplings $\chi^{\alpha\eta}$ defined in eq. (4). The flavor indices α and β can be E, MU, or TAU.
ABS_CHI_ij ARG_CHI_ij	the absolute values and phases of the effective DM–neutrino couplings χ^{ij} expressed in the mass basis ($i, j = 1, 2, 3$). As ABS_CHI_ij, ARG_CHI_ij on one side, and ABS_CHI_$\alpha\beta$, ARG_CHI_$\alpha\beta$ on the other side are not independent, only one of these parameters sets should be used in any given simulaion, with the parameters of the other set set to zero. If parameters from both sets are non-zero, the code will first transform the matrix defined by ABS_CHI_ij, ARG_CHI_ij into the flavor basis and then add the matrix defined by ABS_CHI_$\alpha\beta$, ARG_CHI_$\alpha\beta$. Calls to glbGetOscillationParameters will always return the coupling matrix in the flavor basis, and will set ABS_CHI_ij and ARG_CHI_ij to zero. For scalar DM, the coupling matrix should be symmetric, for vector DM it should be Hermitian. If this requirement is not satisfied, the matrix will be made symmetric / hermitian before using it to compute oscillation probabilities.
ABS_XI_i, ARG_XI_i	the components of the DM polarization 4-vector ξ. The index i runs from 0 to 3. Note that there is some redundancy in these parameters: for scalar DM, they are completely ignored, for unpolarized vector DM, only $\lvert\vec{\xi}\rvert$ matters, and for polarized vector DM, the physically relevant parameters are $\lvert\vec{\xi}\rvert$, ξ^0, and ξ^3. (We always assume the neutrino beam to be oriented along the z-axis.)